Signature Router & Management System - Architecture Documentation

Version: 1.0.0
Date: 2025-11-26
Status: ✅ IMPLEMENTATION READY
Architect: BMAD Architect Agent

📋 Document Index

Esta es la arquitectura completa del **Sistema de Enrutamiento y Gestión de Firmas Digitales** de nivel bancario.

Core Architecture Documents

#	Document	Description	Status
01	System Overview	Visión general, C4 diagrams, bounded contexts	✅ Complete
02	Hexagonal Structure	Package structure, layer responsibilities, patterns	✅ Complete
03	Database Schema	PostgreSQL schema, indexes, security constraints	✅ Complete
04	Event Catalog	Kafka events, Avro schemas, Debezium config	✅ Complete
05	API Contracts	OpenAPI 3.1 specification (REST API)	✅ Complete
06	Resilience Strategy	Circuit breaker, retry, fallback, bulkhead	✅ Complete
07	Observability & Security	Logging, metrics, tracing, auth, encryption	✅ Complete
08	Admin Portal	React SPA architecture and components	✅ Complete

---

🏗 System Architecture Summary

Bounded Context

Un solo bounded context: `Signature Context`

- Enfoque en orquestación de firmas digitales multi-canal
- DDD puro con agregados bien definidos
- Hexagonal architecture para máxima testabilidad

Aggregate Root

```
SignatureRequest (Aggregate Root)
  └─ SignatureChallenge (Entity)
  └─ RoutingRule (Entity - gestión independiente)
  └─ ConnectorConfig (Entity - configuración de providers)
```

Technology Stack

Layer	Technology
Backend	Spring Boot 3 + Java 21
Database	PostgreSQL 15 (JSONB, TDE)
Event Streaming	Kafka + Schema Registry (Avro)
CDC	Debezium (Outbox pattern)
Resilience	Resilience4j (Circuit Breaker, Retry, Bulkhead)
Observability	Prometheus + Grafana + Jaeger/Zipkin
Admin Portal	React 18 + TypeScript + Material-UI
Secrets	HashiCorp Vault

🎯 Key Design Decisions

1. Hexagonal Architecture

- **Why**: Aislamiento total del dominio bancario crítico
- **Benefit**: Testeable sin infraestructura, cambios de providers sin tocar lógica de negocio

2. Outbox + Debezium

- **Why**: Garantía de entrega de eventos (at-least-once)
- **Benefit**: Atomicidad (cambio DB + evento en misma TX), desacoplamiento de Kafka

3. SpEL para Reglas

- **Why**: Expresividad sin código custom, validación previa a persistencia
- **Benefit**: Admins pueden modificar routing sin deployments

4. UUIDv7

- **Why**: Sortable por tiempo, generación distribuida sin coordinación
- **Benefit**: Clustering index eficiente en PostgreSQL

5. Pseudonimización Obligatoria

- **Why**: Compliance GDPR, PCI-DSS
- **Benefit**: Sin PII en logs, eventos, o base de datos

🔄 Signature Flow

```
1. POST /signatures (client)
   └─ StartSignatureUseCase
      ├─ RoutingService.evaluateRoute() → SpEL rules
      ├─ SignatureRequest.createChallenge()
      ├─ SignatureProviderPort.sendChallenge() → Twilio/Push/Voice
      ├─ EventPublisher.publish() → Outbox table
      └─ Return SignatureResponse

2. Debezium CDC
   └─ outbox_event → Kafka (signature.events topic)

3. Consumers
   ├─ Analytics Service
   ├─ Notification Service
   └─ Audit Service
```

```
4. User completes challenge
   └─ PATCH /signatures/{id}/complete
       ├─ SignatureRequest.complete()
       ├─ Store provider_proof (non-repudiation)
       └─ Publish SIGNATURE_COMPLETED event
```

🚨 Resilience Patterns

Fallback Chain

```
PUSH (cheapest) → SMS (reliable) → VOICE (highest success rate) → FAILED
```

Circuit Breaker

- **Threshold**: 50% error rate
- **Wait Duration**: 30s en OPEN state
- **Action**: Provider entra en degraded mode por 5 minutos

Timeouts

Target	Timeout
External HTTP (providers)	5s
Internal HTTP	3s
JDBC	2s
Kafka	1.5s

📊 SLOs (Service Level Objectives)

Metric	Target	Measurement
P99 Latency	< 300ms	End-to-end signature request
Availability	≥ 99.9%	Monthly uptime
Error Rate	< 0.1%	Business logic errors
Data Loss	0%	Event delivery guarantee (Kafka)

🔐 Security Highlights

- **TLS/HTTPS**: All external communication
- **TDE**: Encryption at-rest (PostgreSQL)
- **Vault**: Provider credentials & encryption keys
- **Pseudonymization**: Mandatory for all customer data
- **JWT OAuth2**: Bearer tokens for authentication
- **RBAC**: Role-based access control (Admin, Auditor, Support)
- **Audit Log**: Immutable trail with partition rotation

📈 Observability

Three Pillars

1. **Metrics** (Prometheus + Grafana)
   - Business: `signature.created`, `challenge.sent`, `fallback.rate`
   - Technical: `provider.latency`, `provider.error_rate`
   - SLOs: `signature.duration` (P99), availability
2. **Logs** (Structured JSON + ELK/Loki)
   - MDC context: `traceId`, `signatureId`, `customerId` (tokenized)
   - No PII in logs
3. **Traces** (Jaeger/Zipkin)
   - Distributed tracing across services
   - Propagation to Kafka consumers

🌐 Admin Portal Features

1. **Rule Management**
   - CRUD routing rules with SpEL validation
   - Priority-based ordering
   - Live SpEL syntax checker
2. **Routing Timeline**
   - Visual timeline of challenge attempts
   - Fallback chain visualization

    - Provider responses
3. **Cost Optimization Dashboard**
    - Cost per channel (SMS vs Push vs Voice)
    - Savings calculation
    - Channel distribution
4. **Provider Health Monitor**
    - Error rate tracking
    - Circuit breaker status
    - Degraded mode timeline
5. **Audit Log Viewer**
    - Read-only access for Auditor role
    - Filter by entity, action, date
    - Export to CSV

📦 Deployment Architecture

```
┌─────────────────────────────────────────────────────────────┐
│                    Kubernetes Cluster                        │
│                                                              │
│  ┌───────────────────────────────────────────────────────┐  │
│  │ Signature Engine (Spring Boot)                        │  │
│  │ • Deployment: 3 replicas                              │  │
│  │ • Service: ClusterIP                                  │  │
│  │ • Ingress: /api/v1/**                                 │  │
│  └───────────────────────────────────────────────────────┘  │
│                                                              │
│  ┌───────────────────────────────────────────────────────┐  │
│  │ Admin Portal (React SPA)                              │  │
│  │ • Deployment: 2 replicas (Nginx)                      │  │
│  │ • Service: ClusterIP                                  │  │
│  │ • Ingress: /admin/**                                  │  │
│  └───────────────────────────────────────────────────────┘  │
│                                                              │
│  ┌───────────────────────────────────────────────────────┐  │
│  │ PostgreSQL 15 (StatefulSet)                           │  │
│  │ • Persistent Volume: 100GB                            │  │
│  │ • Backup: Daily snapshots                             │  │
│  └───────────────────────────────────────────────────────┘  │
│                                                              │
│  ┌───────────────────────────────────────────────────────┐  │
│  │ Kafka + Zookeeper (Strimzi Operator)                │   │
│  │ • Cluster: 3 brokers                                     │  │
│  │ • Replication: 3                                         │  │
│  └──────────────────────────────────────────────────────────┘  │
│                                                                  │
│  ┌──────────────────────────────────────────────────────────┐  │
│  │ Debezium Connect                                         │  │
│  │ • CDC connector to Kafka                                 │  │
│  └──────────────────────────────────────────────────────────┘  │
└────────────────────────────────────────────────────────────────┘

External Services:
  • HashiCorp Vault (secrets)
  • Prometheus + Grafana (monitoring)
  • Jaeger (distributed tracing)
  • Twilio API (SMS provider)
```

📝 Implementation Roadmap

F1: Foundation (MVP)

- ✅ Domain models + repositories
- ✅ SMS provider (Twilio) integration
- ✅ Basic routing rules (SpEL evaluation)
- ✅ PostgreSQL persistence
- ✅ Kafka events (basic)
- ✅ REST API (create signature, get status)
- ✅ Admin Portal (rule management)

Estimate: 6-8 semanas

F2: Resilience & Fallback

- ✅ Multi-provider fallback chain
- ✅ Circuit breaker + degraded mode
- ✅ Resilience4j integration (retry, timeout, bulkhead)
- ✅ Provider health monitoring
- ✅ Observability completa (metrics, logs, traces)

Estimate: 4-6 semanas

F3: Multi-Channel

- ✅ Push notification provider
- ✅ Voice call provider
- ✅ Biometrics provider (stub for future)
- ✅ Cost optimization dashboard
- ✅ A/B testing framework (básico)

Estimate: 4-5 semanas

F4: ML & Advanced Features

- ⏳ ML-based routing optimization
- ⏳ Advanced A/B testing
- ⏳ Fraud detection integration
- ⏳ Real-time anomaly detection

Estimate: 8-10 semanas

🚀 Quick Start (for Developers)

Prerequisites

```
# Required
- Java 21
- Maven 3.9+
- Docker + Docker Compose
- Node.js 18+
```

Local Development

```
# 1. Start infrastructure
cd docker
docker-compose up -d  # PostgreSQL, Kafka, Zookeeper, Vault

# 2. Start backend
cd signature-router
mvn spring-boot:run -Dspring.profiles.active=dev

# 3. Start admin portal
cd admin-portal
npm install
npm run dev
```

```
# 4. Access
Backend API: http://localhost:8080
Admin Portal: http://localhost:3000
Swagger UI: http://localhost:8080/swagger-ui.html
```

📚 Additional Resources

- **API Documentation**: [OpenAPI Spec]
- **Database Migrations**: `src/main/resources/db/migration/`
- **Event Schemas**: `src/main/resources/kafka/schemas/`
- **Testing Guide**: `docs/testing-guide.md` (TODO)
- **Deployment Guide**: `docs/deployment-guide.md` (TODO)

👥 Team Contacts

Role	Responsibility
Product Owner	Requirements, backlog prioritization
Architect	System design, technical decisions
Backend Lead	Java implementation, API design
Frontend Lead	Admin Portal (React)
DevOps Lead	Infrastructure, CI/CD, monitoring
QA Lead	Test strategy, automation
Security Lead	Compliance, pen testing, audits

✅ Architecture Review Checklist

✔️ **Domain Model**: Aggregates, Entities, Value Objects defined
✔️ **Hexagonal Architecture**: Clear layer separation
✔️ **Database Design**: Schema, indexes, constraints
✔️ **Event Catalog**: All domain events specified
✔️ **API Contracts**: OpenAPI 3.1 complete

- ✅ **Resilience Patterns**: Circuit breaker, retry, fallback
- ✅ **Security**: TLS, TDE, Vault, pseudonymization
- ✅ **Observability**: Metrics, logs, traces configured
- ✅ **Admin Portal**: UI wireframes and architecture
- ✅ **Testing Strategy**: Unit, integration, E2E tests
- ✅ **Deployment**: Kubernetes manifests ready

📄 License

Architecture Status: ✅ **APPROVED FOR IMPLEMENTATION**

Next Phase: Development Sprint 1 (Foundation)

Last Updated: 2025-11-26 by BMAD Architect Agent